基于级联体素纹理的 VCT 全局光照算法 *

张 菁[1,2,3]，王 鹤[1]，张晓东[1]

(1. 哈尔滨工程大学 计算机科学与技术学院，哈尔滨 150006; 2. 汕头大学 工程学院，广东 汕头 515063; 3. 济南大学 信息科学与工程学院，济南 250022)

摘 要：针对大规模场景的全局光照的渲染往往因为计算量过大而无法满足实时性要求的问题进行了研究，提出一种高性能的体素圆锥追踪（VCT）全局光照算法。算法包括：a)在体素化阶段，提出一种新型的场景光照信息表示结构“级联体素纹理”结构，该结构极大地减少了存储场景的内存消耗而且具有各向异性，在处理大规模场景时具有速度快、质量高的优势；b)在光照计算阶段，基于级联体素纹理结构提出一种改进的圆锥追踪滤波器，能够提高光照注入和反射光计算的效率；c)改进场景的体素化更新策略，进一步提高算法的运行时帧率。实验证明，本算法在降低内存占用空间和提高速度方面均远优于原 VCT 算法，满足了大规模场景中全局光照计算的实时性要求。

关键词：虚拟现实；大规模场景；实时全局光照；级联体素纹理；体素圆锥追踪

中图分类号：TP391.41 doi: 10.19734/j.issn.1001-3695.2018.06.0573

VCT global illumination algorithm based on cascade voxel texture

Zhang Jing[1, 2], Wang He[1], Zhang Xiaodong[1]

(1. *College of Computer Science & Technology, Harbin Engineering University, Harbin* 150006, *China;* 2. *College of Engineering, Shantou University, Shantou Guangdong* 515063, *China*; 3. *School of Information Science & Engineering, Jinan University, Jinan* 250022, *China*)

Abstract: To solve the problem that the global illumination rendering of large scale scene can’t meet the real-time requirement, this paper proposed a voxel cone tracking (VCT) global illumination algorithm. The algorithm includes: a) In the phase of voxelization, this paper proposed a novel scene illumination information representation structure called "cascade voxel textures". This structure can not only greatly reduce the memory consumption of the storage scenes, but also have anisotropy. b) In the phase of illumination calculation, this paper proposed an improved cone tracing filter based on the cascade texture structure. This filter improved the efficiency of illumination injection and reflected light calculation. c) The voxel updating strategy of the scene improved the runtime frame rate of the algorithm. Experimental results show that the proposed algorithm is much better than the original VCT algorithm in reducing the memory and improving the speed. The algorithm meets the real-time requirement of global illumination calculation in large-scale scene.

Key words: virtual reality; large-scale scene; real-time global illumination; cascade voxel texture; voxel cone tracking

0 研究现状

在虚拟世界中，真实的光照是沉浸感的重要来源，其中全局光照是最重要的技术手段。良好的模拟全局光照能够极大的提高场景渲染的真实感。但是，由于复杂场景中的间接光照模拟的难度较高，所以对全局光照算法实时性和准确性的要求也较高。所以，需要一种更快速、近似和自适应的方案解决在大规模复杂场景中实时全局光照的计算问题。目前，主流的全局光照算法分别有 1979 年 Turner Whitted 提出的光线追踪算法[1]及其改进算法[2~4]、1996 年 Jensen[5]提出的光子映射法及其改进算法[6,7],以及 1997 年 Keller 提出的立即辐射度算法[8,9]及其改进算法[10，11]。如今，光子映射算法仍没有达到实时性能，且受到时间闪烁伪影的影响。光线追踪技术和立即辐射度算法虽然实现了在复杂照明情况下的可交互性能，但仍不适用于复杂大规模场景的实时全局光照。

为了满足对于动态全局光照算法的需求，2010 年，Crytek 首次提出动态全局光照算法 Cascaded LPV[12,13]，文中提出使用级联纹理结构保存光照信息，得到了很大的性能提升，该结构对本文有所启示。2011 年，Crassin 等人[14]提出了体素圆锥追踪算法（VCT）,也被称为 VXGI 算法。Cyril 在 VCT 算法中提出一种稀疏体素八叉树的体素存储结构，该结构具有较高的光照计算效率，利用近似计算的思想，完成了对于多种间接光照的模拟，实时性优于 LPV。但其所需的存储空间巨大，且对体素的遍历效率较差。2012 年，Laine 等人[15]提出了改进的稀疏体素八叉树结构（ESVO），一定程度上改善了存储开销，但仍有约 40%的存储空间用于节点的编码。2013 年，Hornung 等人[16]提出了基于 Morton 码的八叉树节点编码（MSVO），该编码方法能够减少部分指针，但其规定的存储方式会造成相同节点之间的冗余存储。2017 年，袁昱纬等人[17]提出的基于稀疏体素有向无环图的光照计算加速结构（SVDAG），极大地减小了体素的存储空间同时加快了渲染过程，但由于该算法是利用合并节点来达到存储目的，导致该方法在处理信息较多的情况时效果不理想，所以对于需要多种信息的全局光照是不利的。2018 年，袁璐[18]通过图

收稿日期：2018-06-03；**修回日期：**2018-07-20 **基金项目：**国家自然科学基金资助项目（51679058）；中国高等专项研究基金资助项目（20132304110018）

作者简介：张菁（1965-），女，安徽怀远人，教授，博士，主要研究方向为虚拟现实、医学图像处理（296582337@qq.com）；王鹤（1994-），女，硕士，主要研究方向为虚拟现实、医学图像处理；张晓东（1993-），男，硕士，主要研究方向为虚拟现实.

形渲染管线计算虚拟点光源并用裁剪后的光源进行延迟着色光照计算，提高了全局光照的效率，但仍不能满足大型动态场景的实时渲染。

综合以上研究现状的结果，针对原 VCT 算法的不足，本文提出针对大规模场景的基于级联体素纹理结构的 VCT 全局光照算法。本文的改进主要包括以下几个方面：

a）通过研究大规模场景结构和人眼观察物体的特点，提出使用级联体素纹理存储体素的方法，通过按层次表示场景，能够极大地减少场景体素规模，提高节点访问效率。并且在级联体素纹理的基础上，改进体素化算法，使得场景体素化速度更快、效率更高，且具有各向异性的性质。

b）全局光照计算复杂的主要原因在于对并行光线的光线追踪任务计算量大、重复性高。本文根据级联纹理存储结构的特点提出一种效率更高的圆锥滤波器。改进的圆锥滤波器能够大幅提高 VCT 全局光照算法的光照计算效率。

c）基于改进的圆锥滤波器的结构特点提出一种适用于级联结构的场景体素更新策略，通过更新策略的改进提高运行时的吞吐量，降低渲染周期。

最后，通过多组对比实验，证明本文提出的方法在保证光照渲染效果的基础上大幅提高了效率，并且减少了内存开销具有更好的效率和真实性。

1　基于级联体素纹理的 VCT 全局光照算法

1.1　级联体素纹理结构

在目前对于全局光照的研究中，大多数方法都采用稀疏八叉树及改进的稀疏八叉树存储结构。但是在稀疏八叉树结构中，体素的数目随着场景规模的增大而增多，因此在大规模场景中会出现体素数量剧增。同时，由于稀疏八叉树的树型结构不便于进行遍历查找且不具有各向异性，体素化速度往往耗时较长。

针对以上问题，本文提出一种新的体素存储结构——级联体素纹理结构。该结构根据人眼对近处景物较敏感，而对远处物体变化敏感度较低的特点，将场景按距离远近分为 n 个不同的区域。用不同大小的互相嵌套的网格来表示这 n 个区域[20]。越近的区域内网格越小，景物被划分的越精细，分辨率越高，越远的区域网格越大，分辨率越低。n 个区域的网格相互嵌套，网格沿着视角的方向延展，呈嵌套正方体的形状，如图 1 所示。通过这种分层表示方式可以达到用较少数目的网格表示大规模场景的目的，避免了随着场景规模增大体素数目激增的情况，同时减少了光线追踪时所需的传播迭代次数。在本文中，为方便表示，将 n 取为 6。

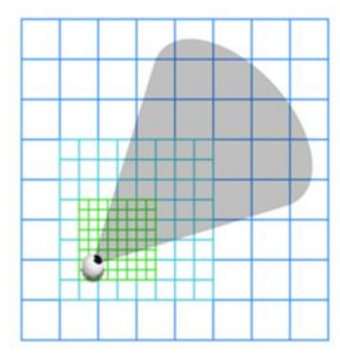


图 1　级联纹理结构示意图（以三层为例）

Fig. 1　Cascading texture structure diagram (taking three layers as example)

级联体素纹理中的最小单位纹素对应场景中的体素。通过这种一一对应关系，将体素中的几何图形相关属性数据[19]存入三维纹理中，将该纹理称为 G-buffer[20]。场景的每项 G-buffer 属性以及光缓冲都需要一张级联纹理保存。为了使结构具有各向异性，本结构存储每个体素 6 个面的信息。级联纹理的结构如图 2 所示。图中 X 轴为体素的六个面，Y 轴为 n 个级联等级，每个纹理单元由 32*32*32 个体素组成，在这里针对本文实验中的图像特点，将 n 取为 6。这样就能以一种合理的方式将场景从近处覆盖到远处。由于本结构中的体素具有各向异性[21]，保证了光照计算的准确性。

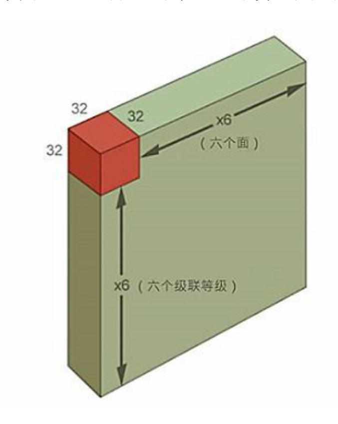


图 2　级联纹理存储结构

Fig. 2　Cascading texture storage structures

这种结构使得在体素之间进行三线性插值变得较为容易，但是需要注意避免取样时误取到其他等级或面的样本而造成计算出错。根据级联体素纹理的存储结构对于体素化过程进行优化，优化后的体素化流程如图 3 所示。

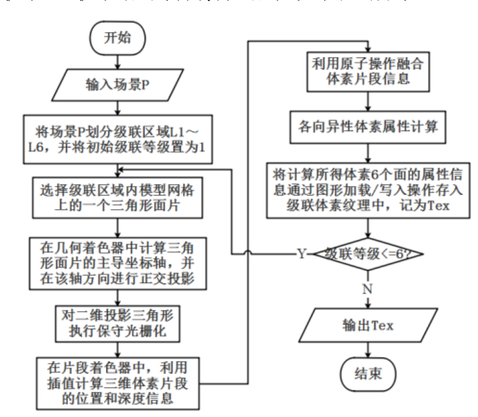


图 3　体素化场景并构建数据结构流程

Fig. 3　Vegan scenarios and building data structure processes

1.2　基于改进圆锥滤波器的圆锥追踪

全局光照的计算通常需要进行大量的光线追踪，虽然光线追踪可以得到很好的渲染效果，但是代价非常昂贵。2011 年，Crassin 提出了圆锥追踪方法将光线在空间和方向上的相干性与抗混叠技术相结合[22, 23]，可以有效解决光线追踪计算量大的问题。

Crassin 提出的圆锥滤波器的尺寸和体素的树型存储结构的 mip 层相对应，每一次使用滤波器收集光照时都需要遍历树型存储结构查找相对应的体素，计算效率较低。为了更有效率地通过圆锥滤波器从场景中收集光亮度，本文根据级联纹理的结构特点，提出一种基于级联体素纹理存储结构的新型圆锥滤波器结构。

改进的结构由 n 组不同尺寸的样本模块按照圆锥滤波器的方向依次排列，结构细节如图 4 所示。

改进的圆锥滤波器不同位置对应不同等级的级联纹理，收集光照时无须进行遍历查找，根据采样的坐标动态的计算

样本尺寸、滤波器直径和级联等级，即可算出该体素在级联纹理中的具体位置，如图 5 所示。接下来对具体方法进行说明。

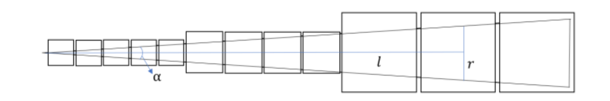


图 4　级联纹理圆锥滤波器

Fig. 4　Cascading texture cone filter

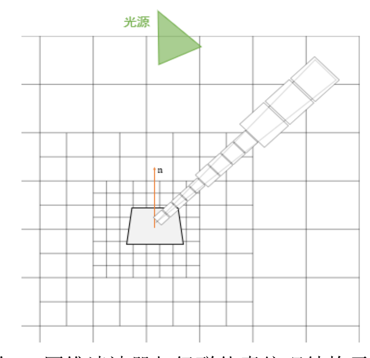


图 5　圆锥滤波器与级联体素纹理结构示意图

Fig. 5　Diagram of texture structure of cone filter and grade conjoined pigment

如图 6 所示，利用渲染方程[24]中的的半球积分[25]可以将球体近似划分为 n 个圆锥体的积分之和。使用体素圆锥追踪来近似它们的增量，通过对结果值加权获得在圆锥原点的累积增量。假设在一个漫反射面上，双向反射比分布函数的所有传入和传出角度都是常量，每个锥中传入的亮度也是常数，则可以将点 x 处的反射光亮度 L_r 重写为

$$L_r(x,\omega_0)=\frac{\rho}{\pi}\sum_{k=1}^{n}L_k(x,\omega_k)W_K \tag{1}$$

其中：ρ 为漫反射系数或反照率[26]；W_k 是一个权重函数；权值和为 2π，用于模拟漫反射；ω_i 的方向为圆锥体的预设方向；圆锥滤波器内的入射光亮度 L_k 是通过将圆锥体分裂成连续的样本元素并进行累积获得的，该过程采用在本文圆锥滤波器中的采样值混合方法。

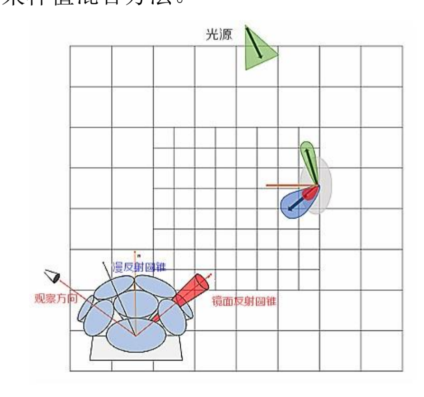


图 6　使用圆锥追踪收集间接光照

Fig. 6　Collect indirect illumination using cone tracking

在光照注入阶段，对于已得到的级联纹理中的每个体素，以其平均法向量为起点，在半圆范围内定义 16 个固定方向的圆锥滤波器用来收集直接光照信息。对于每个滤波器，从样本收集装置到圆锥顶点的距离 l 和角度β决定了样本装置的直径。根据半径和级联等级中体素的宽 Ws，可算出级联等级 d。根据 d、r 和角度 β 即可确定体素位置。

$$d=\log_2(\frac{W_s}{r}) \tag{2}$$

然后对每个圆锥体执行滤波采样获得采样节点值ΣW，最后 16 个圆锥体收集到光亮度累积作为该体素的光亮度，存入 V-buffer 中。通过式（3）动态更新反射光颜色 c 和环境光遮挡值 α。

$$c=\alpha c+(1-\alpha)\alpha_2 c_2,\alpha=\alpha+(1-\alpha)\alpha_2 \tag{3}$$

其中：$c2$、$\alpha2$ 可以直接从当前体素中获得；c 和 α 为 $c2$ 和 $\alpha2$ 的累加值[27]。直接光照被注入到光缓冲中之后，使用分布在法线方向上的 16 个宽锥模拟漫反射，使用分布在反射方向上的单个窄锥模拟镜面反射。将所有采样结果的加权乘以体素的反照率后与直接光照值相加，得到体素的光亮度，包括直接光照以及间接光照的第一次反射。重复各向异性滤波过程，得到间接光照的第二次反射。同时，利用环境光遮挡值 α 和环境光遮蔽项 ao，根据式（4）计算出环境光遮蔽效果。

$$ao=ao+(1-ao)\times a/(1+d\lambda) \tag{4}$$

其中：r 为圆锥体的当前半径；λ 为 $f(r)$随距离的衰减系数。计算多次反射，生成最后的渲染图像。

1.3　基于新结构的改进场景体素更新策略

传统方法在建立场景之初对整个场景进行体素化。在之后的更新过程中将场景分为动态和静态两个部分。当动态部分改变时，将动态部分的体素化结构删除，重新进行体素化。这个过程往往一帧进行一次，造成了计算量的急速增大，降低运行的效率。

因此，针对人眼对近处物体敏感，对远处物体的感知能力下降的特点，在级联体素纹理的基础上提出一种改进的更新方式，对场景中的体素分级更新。根据不同的级联等级 d 设置不同的更新频率 f，f 的计算方法如式（5）所示。级联等级越低，距离观测点越近，则更新频率越高；级别越高，则更新频率越低。此方法降低了系统处理节点频率，提高了运行的吞吐量和效率，降低了计算量，确保了最近的照明反应是以合理的速度更新。

$$f=\log_2 d \tag{5}$$

当更新级联体素纹理不同级别的区域时，若观察者发生了位移，则首先更新级联纹理的中心。对于已更新的中心，更新级联中包括光缓冲和几何缓冲在内的相应数据。对于处于级联区域边缘处的体素，滚动地从下一个级联等级区域中获得相应数据，从而得到边缘处光照的 Mipmap[28]近似值。如果级联级别已移动，还须在级联体素纹理边缘处对新出现的或已经改变的几何模型重新进行体素化。本算法不会对动态对象如人、车辆等填充几何缓冲，因为运动会导致很多重复的体素化工作，不利于全局光照的计算。

基于新结构改进的场景体素更新策略流程如图 7 所示。

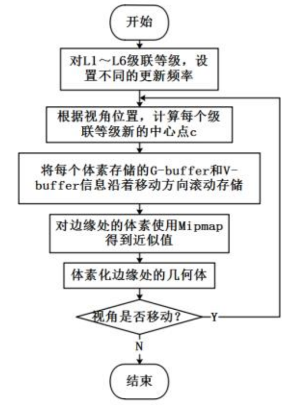


图 7　级联纹理的更新流程

Fig. 7　Update process for cascading textures

综上所述，本文提出的基于级联体素纹理结构的 VCT

全局光照算法算法，详细步骤如下所示：

a)构建级联体素纹理，划分 n 个等级区域并设置更新频率。

b)分别对每个区域进行体素化并计算各向异性，将各区域数据存入级联体素纹理的相应位置。

c)判断观察体是否改变，是则执行 d)，否则执行 e)。

d)更新观察体位置和级联等级区域，体素化新场景部分。

e)从光源处渲染场景，计算直接光照，并将光照信息存入级联体素纹理。

f)使用改进的圆锥追踪滤波器收集间接光照，计算漫反射、镜面反射等，并将光照信息存入级联体素纹理。

g)执行最终渲染，更新屏幕，转 c)。

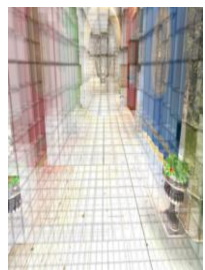
2　实验结果与分析

为了验证本文设计实现的全局光照改进算法适用于大规模场景实时光照渲染且具有较大的性能提升，分别选取静态和动态两类场景进行测试。静态场景为康奈尔盒子，内部分别放置斯坦福大学公开数据集中提供的 Stanford dragon、Stanford bunny 和 Stanford Happy Buddha。动态场景选用由 Frank Meinl 创建的 Sponza 动态场景和由 Marko Dabrovic 创建的 Sibenic Cathedral 动态场景。首先对基于级联体素纹理结构的体素化过程从体素规模、速度和内存消耗三个方面进行性能对比实验，证明本文提出的级联纹理结构的优势；然后对比改进的圆锥追踪方法与原方法在各光照阶段的速度，证明改进的圆锥追踪方法的优越性；最后通过对比四种光照算法在动态更新时的帧率和内存消耗，证明本文提出更新策略在性能上有较大的提升，并通过四种光照算法的绘制效果的对比证明本文提出的算法有较强的实用性。

2.1　基于级联体素纹理结构的体素化性能对比实验

为了证明级联体素纹理在大规模场景中体素化阶段的优越性，本章分别对级联体素纹理、三维纹理、稀疏八叉树、文献[16]中的 MSVO 和[17]中提到的 SVDAG 结构进行实验。在体素规模、体素化时间和存储开销三方面的性能进行测试对比。

为了更直观地说明本文的体素结构，在开始实验之前，利用体素可视化的对比图像进行简单说明。






(a)级联体素纹理　　(b)稀疏八叉树　　(c)三维纹理

图 8　体素可视化

Fig. 8　Voxel visualization

图 8 (a)为使用本文提出的级联体素纹理结构表示的场景，可以清晰地看到，场景中的体素是分层排列的，近处的体素精细，远处的体素体积较大，覆盖区域更广；(b)为稀疏八叉树结构表示的场景，不论远处还是近处，体素的排布都很精细，虽然这样可以获得较好的渲染效果，但体素规模过多导致计算量过大、耗时过长，不利于实时应用；(c)为使用三维纹理结构表示的场景，三维纹理覆盖了整个场景，即使相应位置没有体素也预留了存储空间，造成空间的极大浪费，光照计算效率也较低。

2.1.1 体素规模对比

表 1 为不同结构对于静态场景体素化规模的对比。表 2 为不同结构对于动态场景体素化规模的对比。从表 1 和 2 中可以看出，由于三维纹理没有区分区域，所以在各分辨率情况下体素总量规模都是最大的。稀疏八叉树表示的场景体素规模远低于三维纹理，但高于级联体素纹理；而 MSVO 和 SVDAG 两种结构比稀疏八叉树有一定程度上的提高，但对于大规模场景来说依然结果不理想。

从表中可以看出，分辨率较低时，稀疏八叉树和级联体素纹理的体素规模相差不大；但随着分辨率的增大，它们之间的差距也越来越大。这是因为在小规模场景中，级联纹理将场景分层表示的优势变小；而随着分辨率的提高，这种优势逐渐显现出来。级联体素纹理将场景按层次划分，相邻层之间体素大小呈倍数关系线性递增。在不同分辨率情况下，场景的表示仅需增加层数；在层数一定的情况下，其体素化的规模也是相同的。因为这样的结构特点使得级联体素纹理在表示大规模场景时，能够在保证渲染效果的基础上极大地减少体素的数量。

表 1　静态场景体素规模对比　　/10^6 个

Table 1　Scale comparison of voxels in static scenes

	Bunny		Dragon		Buddha	
分辨率	128^3	256^3	128^3	256^3	128^3	256^3
三维纹理	2.097	16.78	2.097	16.78	2.097	16.78
稀疏八叉树	0.189	3.604	0.113	2.364	0.166	3.215
MSVO	0.176	3.455	0.101	2.178	0.151	2.986
SVDAG	0.133	2.562	0.891	1.855	0.125	2.411
级联体素纹理	0.098	0.131	0.098	0.131	0.131	0.164

表 2　动态场景体素规模对比　　/10^6 个

Table 2　Scale comparison of voxels in dynamic scenes

	Sponza		Sibenic Cathedral	
分辨率	256^3	512^3	256^3	512^3
三维纹理	16.78	134.2	16.78	134.2
稀疏八叉树	3.135	6.032	2.916	5.489
MSVO	2.885	5.766	2.598	5.247
SVDAG	2.445	4.705	2.225	4,282
级联体素纹理	0.131	0.164	0.131	0.164

2.1.2 体素化速度对比

表 3 和 4 显示了不同方案的时间开销情况。通过观察数据可以发现，由于稀疏八叉树的数据结构逻辑复杂，不便于实现快速实现构建、访问和遍历等基本数据操作，其体素化时间要高于三维纹理。同样地，基于稀疏八叉树改进的 MSVO 和 SVDAG 也存在相同的问题，且时间消耗要略高于稀疏八叉树；而级联体素纹理的分层结构便于遍历及查找，极大地缩短了体素化的过程，使其与 GPU 协作更顺畅。基于级联体素纹理结构改进的体素化算法在静态场景和动态场景中，速度均有大幅提高。

表 3　静态场景体素化时间对比　　/ms

Table 3　Voxelization time comparison in static scene

	Bunny		Dragon		Buddha	
分辨率	128^3	256^3	128^3	256^3	128^3	256^3
三维纹理	14.96	43.22	15.36	42.75	16.16	45.65
稀疏八叉树	56.17	96.41	46.81	80.34	60.85	104.4
MSVO	53.26	92.34	44.29	76.67	58.54	101.3
SVDAG	62.71	135.51	53.24	109.44	72.86	166.0
级联体素纹理	5.179	6.872	4.316	5.727	5.611	7.445

表 4 动态场景体素化时间对比 /ms

Table 4 Voxelization time comparison of dynamic scenes

	Sponza		Sibenic Cathedral	
分辨率	256^3	512^3	128^3	256^3
三维纹理	51.29	230.1	52.11	233.8
稀疏八叉树	88.48	280.8	98.19	314.5
MSVO	86.96	275.4	96.78	309.7
SVDAG	105.1	336.3	115.1	384.4
级联体素纹理	6.889	8.549	7.716	9.575

本文改进的体素化算法增加了对于体素的各向异性的计算，但是并没有执行多余的绘制，仍然满足了时间上的要求。综上所述，级联体素纹理方案的体素化速度在大小场景中均有较大优势，能够满足在大规模复杂场景中对于动态全局光照渲染的体素化需求。

2.1.3 内存消耗对比

在内存开销方面，表 5 和 6 比较了不同结构的内存开销情况。由于像素点的各相关信息所占空间情况具有相似性，为了去除多种信息对于实验的干扰，在本实验中仅以透明度属性所占用的空间为例进行统计说明，每个节点有一个透明度属性信息，每个透明度属性信息占用一个字节空间。

本文中提出的级联体素纹理具有各向异性，能够存储体素六个面的信息，而其他几种方法仅存储体素一个面的信息，但由于级联体素纹理的分层结构，其存储开销明显小于其他方法。随着分辨率的提高，三维纹理、稀疏八叉树和 MSVO 所消耗的内存大小以指数倍上升，SVDAG 方法消耗的内存比稀疏八叉树有非常明显的下降，但依然是以指数倍增加的，这样的增加速度在大规模场景中会带来极大的消耗；而级联体素纹理是以线性关系增加，对于场景规模较小的静态场景，级联体素纹理的内存开销约是 SVDAG 的 1/3，对于场景规模较大的动态场景，级联体素纹理的内存开销约是 SVDAG 的几十分之一。可以明显地看出，级联体素纹理结构大规模场景中表现出色、优势明显，能够大幅节省内存消耗。

表 5 静态场景体素化时间对比 /MB

Table 5 Voxelization time comparison in static scene

	Bunny		Dragon		Buddha	
分辨率	128^3	256^3	128^3	256^3	128^3	256^3
三维纹理	2	16	2	16	2	16
稀疏八叉树	1.584	12.67	1.32	10.56	1.726	13.21
MSVO	1.679	13.44	1.79	11.98	1.878	14.43
SVDAG	0.951	7.602	0.924	3.392	1.295	9.247
级联体素纹理	0.56	0.75	0.56	0.75	0.56	0.75

表 6 动态场景体素化时间对比 /MB

Table 6 Voxelization time comparison of dynamic scenes

	Sponza		Sibenic Cathedral	
分辨率	256^3	512^3	256^3	512^3
三维纹理	16	128	16	128
稀疏八叉树	6.06	48.48	6.908	54.29
MSVO	7.07	50.87	7.106	56.86
SVDAG	4.545	35.98	5.388	43.35
级联体素纹理	0.75	0.94	0.75	0.94

上述实验证明本文提出的基于级联体素纹理的体素化方案在体素规模、体素化速度、内存消耗三方面性能都非常优秀。尤其是在大规模场景中，极大地减少了表示场景的体素数量，加快了体素化的时间以及节省了内存消耗，并且具有各向异性体素表示的优势。这表明本结构不仅能够减少全局光照的计算量，而且有利于光照效果的绘制。

2.2 改进的圆锥追踪方法在各光照阶段的性能对比实验

圆锥追踪是原 VCT 算法（即 VXGI 算法）中独创的一种较为优秀的追踪方式。本节通过对比本文改进的圆锥追踪方法和 VXGI 算法中的圆锥追踪方法在各光照注入和圆锥追踪阶段所消耗的时间，说明本文提出的改进方法具有更好的性能。

表 7 为 VXGI 算法与本文提出的改进算法在实现每个全局光照效果时渲染每帧所用时间对比。在动态场景中，VXGI 算法体素动态交互更新大约每帧需要 15.5 ms，全部光照效果共计需要 36.7 ms，此时平均运行帧率约为 20 fps，在没有镜面反射时共需 14.2 ms，此时平均运行帧率约为 30 fps。本文算法动态交互更新大约每帧需要 2.0 ms，全部光照效果共计需要 26.5 ms，此时平均运行帧率约为 36 fps，在没有镜面反射时共需 11.3 ms，此时平均运行帧率约为 43 fps，这说明本文提出的改进圆锥滤波器在级联纹理的场景中效果发挥出色，在光照注入和圆锥追踪阶段计算速度更快，使得全局光照计算的效率得到明显提升。

表 7 各光照计算过程的时间消耗 /ms

Table 7 Time consumption of each illumination calculation process

	VXGI 算法	本文提出的算法
光栅化	1.0	1.0
光照注入	16.5	12.1
直接光照	3.0	1.8
漫反射	9.2	6.5
镜面反射	8.0	5.2
总计	36.7	26.5

2.3 基于级联体素纹理结构改进的更新策略性能测试

本实验使用 Sponza 动态场景和 Sibenic Cathedral 动态场景，从运行帧率、内存消耗廉各方面方面测试本文算法，并将实验结果与 LPV、PM 和 VXGI 全局光照算法对比，全面地佐证结论的真实与可靠性。

2.3.1 运行帧率对比

图 9 为原 VCT、LPV、PM、未改进更新策略的本文算法和改进了更新策略的本文算法的运行帧率对比折线图。从图中可以看出，同样是本文提出的算法，改进更新策略后运行帧率有了较大的提高，这说明本文提出的改进更新策略在速度的提升上有较好的效果。同时，即使是未改进更新策略之前的本文算法，速度也略优于 VXGI 算法。VXGI 算法和本文算法的平均运行帧率已经达到实时渲染的标准，而 LPV 和 PM 算法稍差，帧率只能维持在 5~20 fps/s，勉强能够实时交互但卡顿严重，甚至给操作者带来晕眩的症状。本文算法在光照计算的过程中采用了效率更高的圆锥追踪方式和改进的体素更新策略，因此与 VXGI 相比，本文算法的运行帧率平均高出约 10 帧。这说明本文基于级联体素纹理改进的场景体素更新策略更适合大规模场景，使得算法整体的运行速度更快，计算更轻量。

2.3.2 内存消耗对比

本文算法使用改进的体素化更新策略，在运行阶段并没有对整个场景即所有的级联等级区域重新体素化，只是根据不同的频率更新不同的级联区域内的场景以及边界处的新场景，所以进一步减少了内存消耗。在不会对 PC 造成较高负载的前提下，以较小的内存消耗换取了极大的效率提升，保证了良好的光照效果。图 10 为 VXGI 算法、LPV、PM、未改进更新策略的本文算法和改进了更新策略的本文算法的内存消耗与时间的关系。通过未改进的本文算法和改进过后的

本文算法对内存消耗的对比可以证明，改进的更新策略能够有效地降低内存消耗。

从图中可知，PM 算法使用光子图计算全局光照，内存消耗最低；LPV 使用分层的几何信息网格表示光子，占用内存也较小，但两者的速度较差，不适用于复杂场景的实时光照计算；VXGI 和本文算法将场景体素化，体素数据使得内存消耗较高。与 VXGI 算法相比，本文算法的内存消耗仅为 VXGI 的 40%，但效果与 VXGI 非常相近。

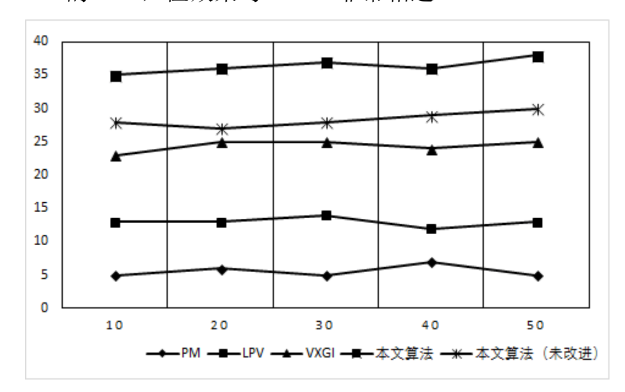


图 9　四种全局光照算法运行帧率对比/fps

Fig. 9　Frame rate comparison of four global illumination algorithms/fps

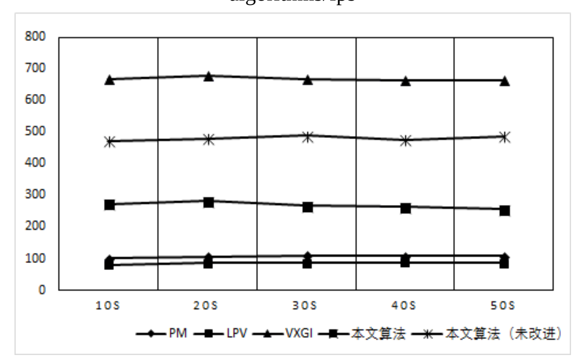


图 10　四种全局光照算法内存消耗对比/MB

Fig. 10　Comparisons of memory consumption of four global illumination algorithms/MB

2.4　四种全局光照算法渲染效果对比实验

由于间接光照的渲染难度远高于直接光照，所以渲染效果的真实性主要取决于对于间接光照的模拟。PM 算法在这方面效果较差，不具有对比意义。所以在本实验中使用 Sponza 动态场景，对比 LPV、VXGI 算法和本文提出的全局光照算法的渲染效果，全面地佐证结论的真实与可靠性。图 11 为 VXGI、LPV 和本文算法的渲染效果对比。左上为 VXGI 算法，右上为本文算法；左下为 LPV 渲染效果，右下为光线追踪模拟出的渲染效果，是当前全局光照领域能够模拟出的最接近真实光照的效果，也是衡量全局光照渲染效果的标尺。

通过对比发现，VXGI 算法和本文算法的渲染效果相差无几，都与真实场景比较接近，实现了较好的视觉质量。虽然本文提出的算法在离观察者很远处使用了体积较大的体素进行近似渲染，但从图中可以看出，即使在很远处渲染效果也非常接近真实效果。因此，可以得到结论：本文提出的基于级联体素结构的全局光照算法，在保证渲染效果的前提下极大地改善了效率和内存消耗，满足了大规模复杂场景对实时的全局光照绘制的要求。

利用本文提出的算法分别对漫反射、镜面反射和环境光遮蔽进行实验，通过效果细节图说明本文算法的实用性。

图 12 的左、右两幅图分别为无漫反射和有漫反射的光照绘制效果。右图通过累加多个漫反射锥的亮度采样值得到漫反射效果后，柱子两侧皆清晰可见，更贴近现实中的光照环境。

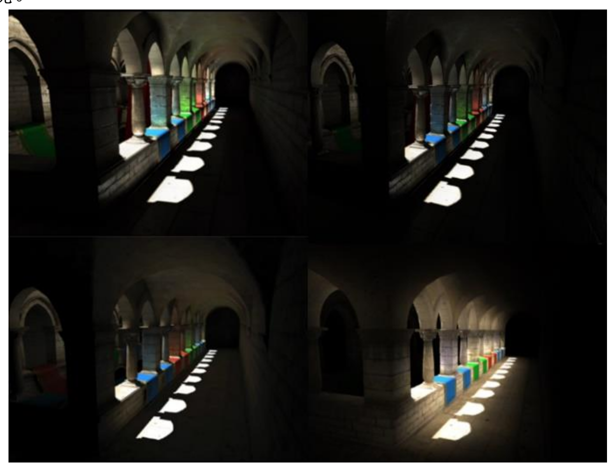


图 11　Sponza 场景渲染效果对比

Fig. 11　Comparison of rendering effects of Sponza scenes

图 13 的左、右两幅图分别为无镜面反射和有镜面反射的光照绘制效果。镜面反射是通过追踪一个镜面反射锥实现的。通过材质的光滑程度确定镜面反射的强度。

通过以上细节图的展示可以看出，本文算法实现的全局光照符合现实规律，渲染效果自然，光线明暗合理，能够使渲染效果更贴近现实的光照效果。

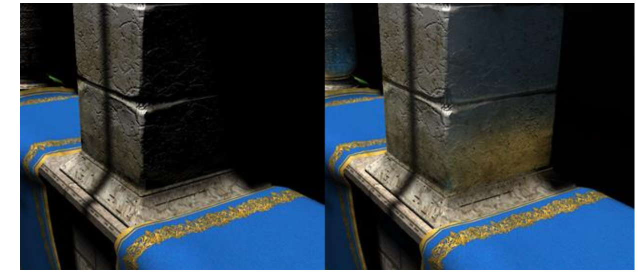


图 12　漫反射效果

Fig. 12　Diffuse reflection effect



图 13　镜面反射效果

Fig. 13　Specular reflection effect

通过以上细节图的展示可以看出，本文算法实现的全局光照符合现实规律，渲染效果自然，光线明暗合理，能够使渲染效果更贴近现实的光照效果。

3　结束语

随着虚拟现实在军事、商业、教育、娱乐等领域不断深入的应用，全局光照成为了真实感模拟、交互、计算机视觉中图像分析以及增强现实中图像融合等技术的重要组成部分。本文对全局光照技术研究成果如下：首先，提出一种在大规模场景中效率较高的体素存储结构——级联体素纹理，它在大规模场景中获得了时间和空间上的性能大幅提升，此外，基于级联体素纹理，改进了原体素化算法流程，使得场

景的体素化效率更高且具有各向异性；其次，提出一种改进的圆锥追踪滤波器，该滤波器能够更好地适应级联体素纹理结构，在滤波时具有高准确率和速度快的优点；最后，通过分析大规模场景结构特点以及人眼感知世界的规律，找出一种更适应需求且更真实自然的体素更新方案，使得算法在速度上得到明显提升。本文通过多组实验证明了本文算法在模拟大规模场景中的直接光照、漫反射、镜面反射等光照效果时取得了较真实的渲染效果，并且效率较高，具有较强的实用性。相对于目前已有的算法，本文提出的算法解决了对于大规模场景的全局光照渲染速度较差的难题，对该技术在各个领域的应用有较大的现实意义和指导意义。

参考文献：

[1] 大野義夫. An improved illumination model for shaded display [J]. Ipsj Magazine, 1981, 22 (2): 22-26.

[2] Calazan R M. Parallel ray tracing for underwater acoustic predictions [J]. Computational Science and Its Applications, 2017, 18 (5): 53-59.

[3] Kao C C, Miao Y T, Hsu W C. A pipeline-based ray-tracing runtime system for HSA-compliant frameworks [J]. IEEE Trans on Multimedia, 2017, 19(11): 2450-2462.

[4] Shivaraju S, Pudur G. Splay thread cooperation on ray tracing as a load balancing technique in speculative parallelism and GPGPU [J]. International Arab Journal of Information Technology, 2018, 15 (1): 167-176.

[5] Jensen H W. Global illumination using photon maps[M]. Vienna：Springer, 1996: 21-30.

[6] Guzek K, Napieralski P. Efficient rendering of caustics with streamed photon mapping [J]. Bulletin of the Polish Academy of Sciences Technical Sciences, 2017, 65 (3): 361-368.

[7] Dong Zong. Pacific conference on computer graphics and applications [Z]. Hawaii:IEEE Computer Society, 2007: 77-86.

[8] Goral C M, Torrance K E, Greenberg D P, *et al*. Modeling the interaction of light between diffuse surfaces [J]. ACM SIGGRAPH Computer Graphics, 1984, 18(3): 213-222.

[9] De Bonet J S. Multiresolution sampling procedure for analysis and synthesis of texture images [C]// Proc of the 24th Annual Conference on Computer Graphics and Interactive Techniques.[S.l.]:ACM Press/Addison-Wesley Publishing Co, 1997: 361-368.

[10] Ritschel T, Grosch T, Kim M H. Imperfect shadow maps for efficient computation of indirect illumination [J]. ACM Trans on Graphics, 2008, 27 (5): 1-8.

[11] Hollander M, Ritschel T, Eisemann E, *et al*. ManyLoDs: parallel many-view level-of-detail selection for real-time global illumination [C] //Proc of Eurographics Conference on Rendering. [S.l.]:Eurographics Association, 2011: 1233-1240.

[12] Kaplanyan A. Light propagation volumes in CryEngine 3[J]. SIGGRAPH, 2009, 12 (3): 11-18.

[13] Kaplanyan A, Dachsbacher C. Cascaded light propagation volumes for real-time indirect illumination [C]//Proc of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games. New York: ACM Press, 2010: 99-107.

[14] Crassin C, Neyret F, Sainz M, *et al*. Interactive indirect illumination using voxel cone tracing [C]//Proc of Symposium on Interactive 3D Graphics and Games. New York: ACM Press, 2011: 207-207.

[15] Laine S, Karras T. Efficient sparse voxel octrees. [J]. IEEE Trans on Visualization & Computer Graphics, 2011, 17 (8): 1048-1059.

[16] Hornung A, Kai M W, Bennewitz M, *et al*. OctoMap: an efficient probabilistic 3D mapping framework based on octrees [J]. Autonomous Robots, 2013, 34 (3): 189-206.

[17] 袁昱纬，全吉成，吴晨，等. 基于稀疏体素有向无环图的光照计算加速结构 [J]. 光学学报，2017, 37 (8): 259-272. (Yuan Yuwei, Quan Jicheng, Wu Chen, *et al*. Light computing accelerated structure based on sparse volume prime directed acyclic graph [J]. Journal of Optics, 2017, 37 (8): 259-272.)

[18] 袁璐. 基于立即辐射度的实时全局光照算法 [J]. 现代计算机:专业版，2018 (2): 24-28. (Yuan Lu. Real-time global illumination algorithm based on immediate radiance [J]. Modern Computer:Professional, 2018 (2): 24-28.)

[19] Zou Changjun, Yin Yong, Jing Qianfeng. Real-time fluid simulation with complex boundary based on slice voxelization method[C]// Proc of Asian Simulation Conference. Berlin: Springer, 2017: 319-326.

[20] Crassin C, Neyret F, Sainz. Interactive indirect illumination using voxel-based cone tracing: an insight [J]. Computer Graphics Forum, 2011, 30 (7): 1921-1930.

[21] Blessing R H. An empirical correction for absorption anisotropy [J]. Acta Crystallographica Section A Foundations of Crystallography, 2014, 51 (1): 33-38.

[22] Martin T, Tan T S. Anti-aliasing and continuity with trapezoidal shadow maps [C]//Proc of the 15th Eurographics Workshop on Rendering Techniques. 2004: 153-160.

[23] Franke T A. Delta voxel cone tracing [C]//Proc of IEEE International Symposium on Mixed and Augmented Reality. Piscataway, NJ: IEEE Press, 2014: 39-44.

[24] Kajiya J T. The rendering equation [J]. ACM Computer Graphics, 1986, 20 (4): 143-150.

[25] Chuah S P, Cheung N M, Yuen C. Layered coding for mobile cloud gaming using scalable blinn-phong lighting [J]. IEEE Trans on Image Process, 2016, 25 (7): 3112-3125.

[26] 《中国百科大辞典》总委员会. 中国百科大辞典 (普及版) [M]. 北京：中国大百科全书出版社, 2005. (Chairman of the General Committee of the Chinese Department of General Dictionary. China encyclopedia dictionary (popular version) [M]. Beijing. China Encyclopedia Press, 2005.)

[27] Chang H R. Energy accumulation and emanation at low latitudes, part III: forward and backward accumulation. [J]. Journal of Atmospheric Sciences, 1995, 52 (13): 2384-2403.

[28] Tanner C C, Migdal C J, Jones M T. The clipmap: a virtual mipmap [C]// Proc of the 25th annual conference on Computer graphics and interactive techniques. New York: ACM Press,, 1998: 151-158.